

Lecture 16: 14 March, 2024

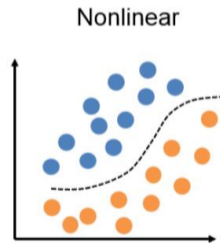
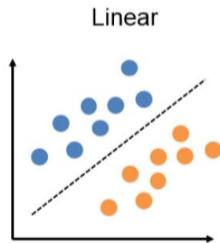
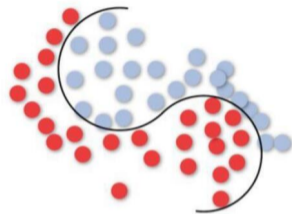
Madhavan Mukund

<https://www.cmi.ac.in/~madhavan>

Data Mining and Machine Learning
January–April 2024

A geometric view of supervised learning

- Think of data as points in space
- Find a separating curve (surface)
- Separable case
 - Each class is a connected region
 - A single curve can separate them
- Simplest case — linearly separable data
- Dual of linear regression
 - Find a line that passes close to a set of points
 - Find a line that separates the two sets of points



Perceptron algorithm

(Frank Rosenblatt, 1958)

- Each training input is (x_i, y_i) , where $x_i = \langle x_{i_1}, x_{i_2}, \dots, x_{i_n} \rangle$ and $y_i = +1$ or -1
- Need to find $w = \langle w_0, w_1, \dots, w_n \rangle$
 - Recall $x_{i_0} = 1$, always

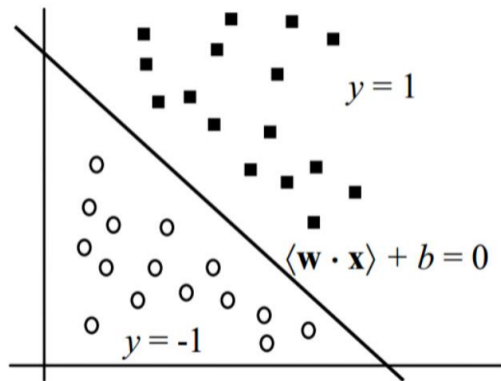
Initialize $w = \langle 0, 0, \dots, 0 \rangle$

While there exists x_i, y_i such that

$y_i = +1$ and $w \cdot x_i < 0$, or

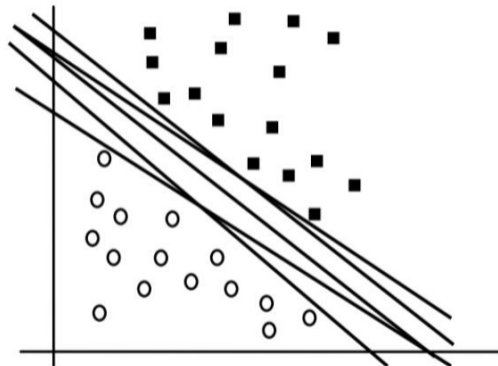
$y_i = -1$ and $w \cdot x_i > 0$

Update w to $w + x_i y_i$



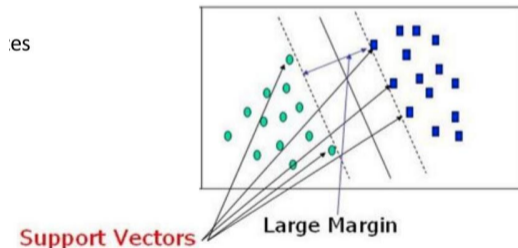
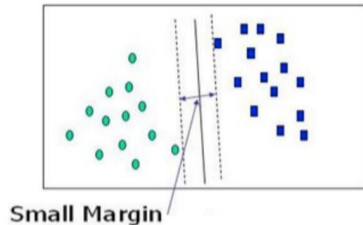
Linear separators

- Perceptron algorithm is a simple procedure to find a linear separator, if one exists
- Many lines are possible
 - Does the Perceptron algorithm find the best one?
 - What is a good notion of “cost” to optimize?



Margin

- Each separator defines a margin
 - Empty corridor separating the points
 - Separator is the centre line of the margin
- Wider margin makes for a more robust classifier
 - More gap between the classes
- Optimum classifier is one that maximizes the width of its margin
- Margin is defined by the training data points on the boundary
 - Support vectors



Finding a maximum margin classifier

- Recall our original linear classifier

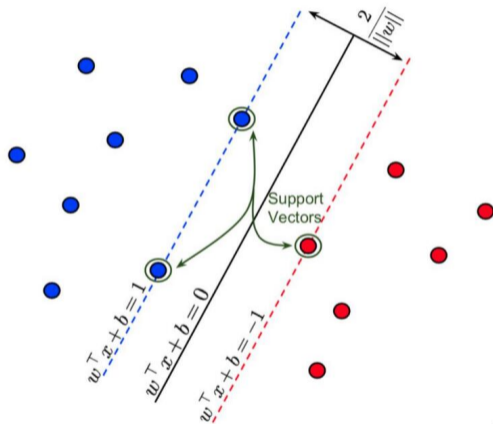
$w_1x_1 + w_2x_2 + \dots + w_nx_n + b > 0$, classify
yes, $+1$

$w_1x_1 + w_2x_2 + \dots + w_nx_n + b < 0$, classify
no, -1

- Scale margin so that separation is 1 on
either side

$w_1x_1 + w_2x_2 + \dots + w_nx_n + b > 1$, classify
yes, $+1$

$w_1x_1 + w_2x_2 + \dots + w_nx_n + b < -1$, classify
no, -1

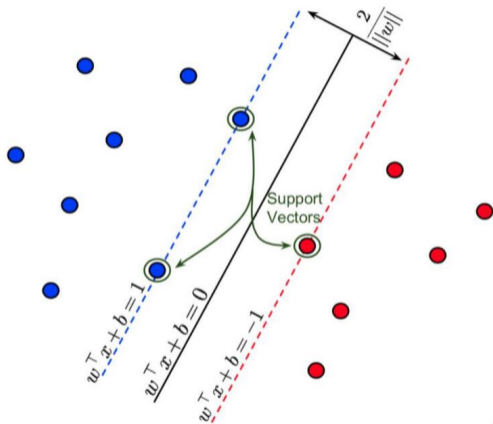


Finding a maximum margin classifier

- Scale margin so that separation is 1 on either side
 $w_1x_1 + w_2x_2 + \dots + w_nx_n + b > 1$, classify yes, +1
 $w_1x_1 + w_2x_2 + \dots + w_nx_n + b < -1$, classify no, -1

- Using Pythagoras's theorem, perpendicular distance to nearest support vector is $\frac{1}{|w|}$, where

$$|w| = \sqrt{w_1^2 + w_2^2 + \dots + w_n^2}$$



Optimization problem

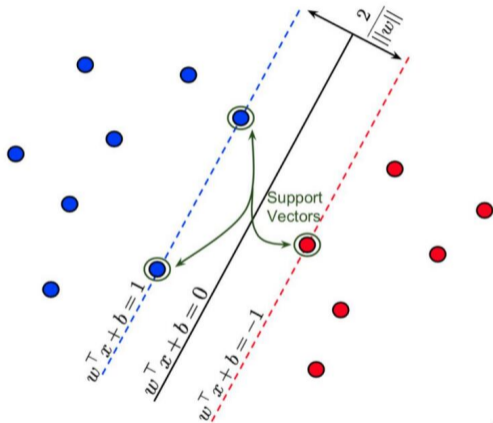
- Want to maximize the overall margin $\frac{2}{|w|}$

- Equivalently, minimize $\frac{|w|}{2}$

- Also, w should classify each (x_i, y_i) correctly

$$w_1x_1^i + w_2x_2^i + \cdots w_nx_n^i + b > 1, \\ \text{if } y_i = 1$$

$$w_1x_1^i + w_2x_2^i + \cdots w_nx_n^i + b < -1, \\ \text{if } y_i = -1$$



Optimization problem

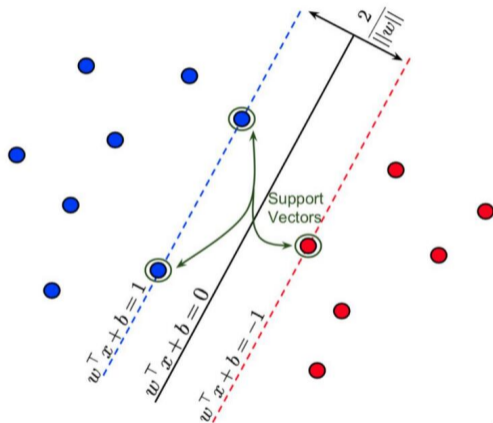
$$\text{Minimize } \frac{|w|}{2}$$

Subject to

$$w_1 x_1^i + w_2 x_2^i + \cdots + w_n x_n^i + b > 1, \text{ if } y_i = 1$$

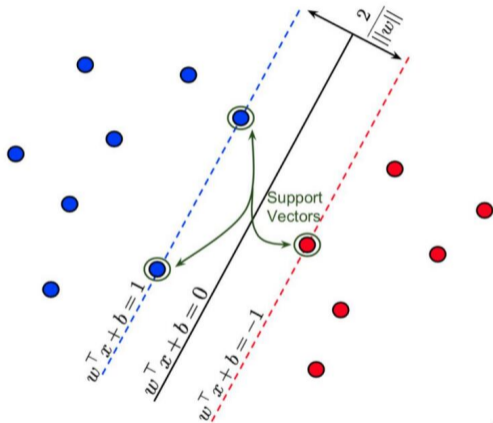
$$w_1 x_1^i + w_2 x_2^i + \cdots + w_n x_n^i + b < -1, \text{ if } y_i = -1$$

- The constraints are linear
- The objective function is not linear
 $|w| = \sqrt{w_1^2 + w_2^2 + \cdots + w_n^2}$
- This is a **quadratic optimization problem**, not linear programming



Solution to optimization problem

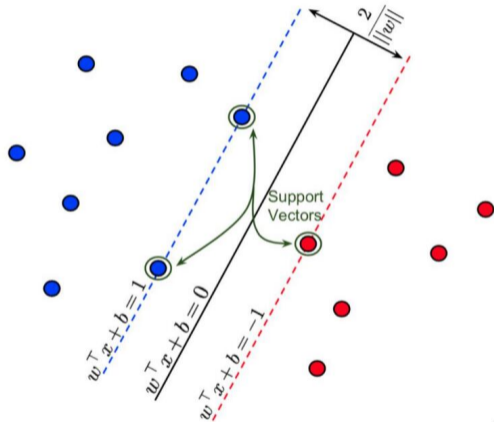
- Convex optimization theory
- Can be solved using computational techniques
- Solution expressed in terms of Lagrange multipliers $\alpha_1, \alpha_2, \dots, \alpha_N$, one multiplier per training input
- α_i is non-zero iff x_i is a support vector
- Final classifier for new input z
$$\text{sign} \left[\sum_{i \in \text{sv}} y_i \alpha_i (x_i \cdot z) + b \right]$$
- sv is set of support vectors



Support Vector Machine (SVM)

$$\text{sign} \left[\sum_{i \in \text{sv}} y_i \alpha_i (x_i \cdot z) + b \right]$$

- Solution depends only on support vectors
 - If we add more training data away from support vectors, separator does not change
- Solution uses dot product of support vectors with new point
 - Will be used later, in the non-linear case



The non-linear case

- Some points may lie on the wrong side of the classifier
- How do we account for these?
- Add an error term to the classifier requirement
- Instead of

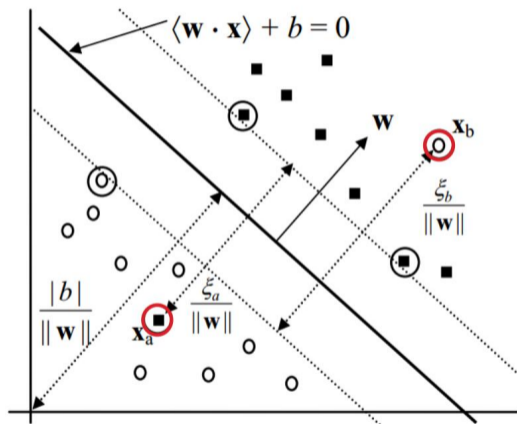
$$w \cdot x + b > 1, \text{ if } y_i = 1$$

$$w \cdot x + b < -1, \text{ if } y_i = -1$$

we have

$$w \cdot x + b > 1 - \xi_i, \text{ if } y_i = 1$$

$$w \cdot x + b < -1 + \xi_i, \text{ if } y_i = -1$$

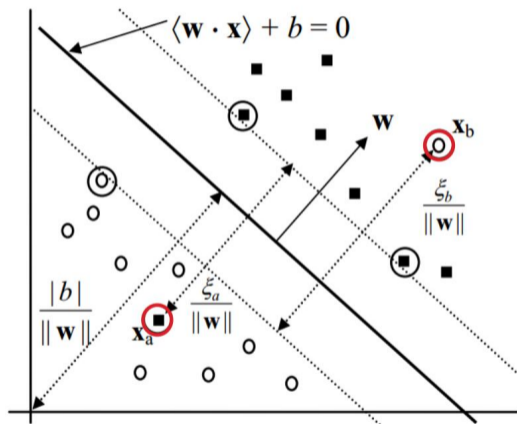


Soft margin classifier

$$w \cdot x + b > 1 - \xi_i, \text{ if } y_i = 1$$

$$w \cdot x + b < -1 + \xi_i, \text{ if } y_i = -1$$

- Error term always non-negative,
- If the point is correctly classified, error term is 0
- **Soft margin** — some points can drift across the boundary
- Need to account for the errors in the objective function
 - Minimize the need for non-zero error terms



Soft margin optimization

$$\text{Minimize } \frac{|w|}{2} + \sum_{i=1}^N \xi_i^2$$

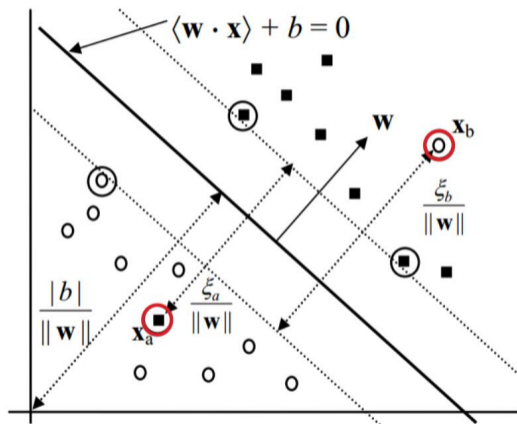
Subject to

$$\xi_i \geq 0$$

$$w \cdot x_i + b > 1 - \xi_i, \text{ if } y_i = 1$$

$$w \cdot x_i + b < -1 + \xi_i, \text{ if } y_i = -1$$

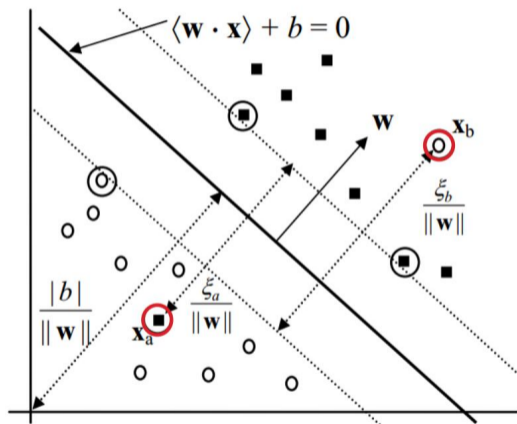
- Constraints include requirement that error terms are non-negative
- Again the objective function is quadratic



Soft margin optimization

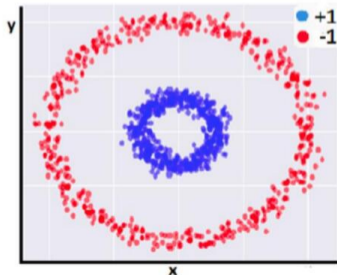
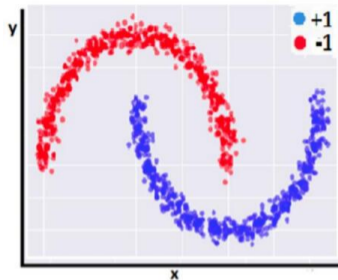
- Can again be solved using convex optimization theory
- Form of the solution turns out to be the same as the hard margin case
 - Expression in terms of Lagrange multipliers α_j
 - Only terms corresponding to support vectors are actively used

$$\text{sign} \left[\sum_{i \in \text{sv}} y_i \alpha_i (x_i \cdot z) + b \right]$$



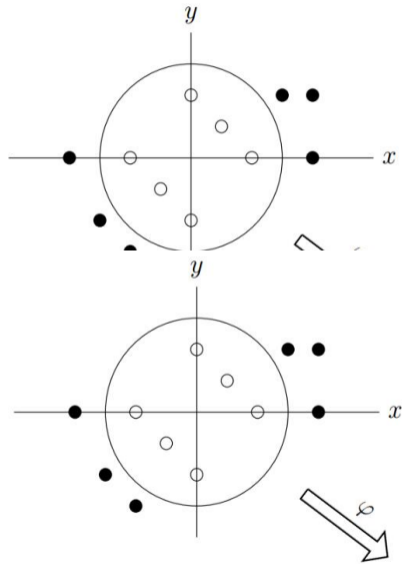
The non-linear case

- How do we deal with datasets where the separator is a complex shape?
- Geometrically transform the data
 - Typically, add dimensions
- For instance, if we can “lift” one class, we can find a planar separator between levels



Geometric transformation

- Consider two sets of points separated by a circle of radius 1
- Equation of circle is $x^2 + y^2 = 1$
- Points inside the circle, $x^2 + y^2 < 1$
- Points outside circle, $x^2 + y^2 > 1$
- Transformation
$$\varphi : (x, y) \mapsto (x, y, x^2 + y^2)$$
- Points inside circle lie below $z = 1$
- Point outside circle lifted above $z = 1$



SVM after transformation

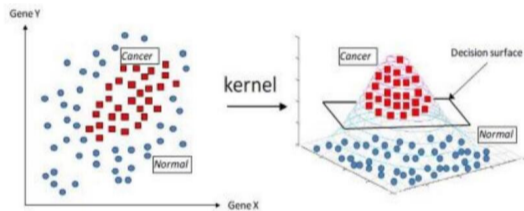
- SVM in original space

$$\text{sign} \left[\sum_{i \in \text{sv}} y_i \alpha_i (x_i \cdot z) + b \right]$$

- After transformation

$$\text{sign} \left[\sum_{i \in \text{sv}} y_i \alpha_i (\varphi(x_i) \cdot \varphi(z)) + b \right]$$

- All we need to know is how to compute dot products in transformed space



Dot products

- Consider the transformation

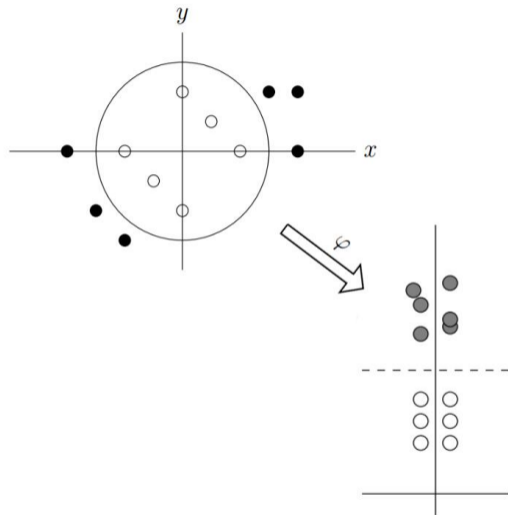
$$\varphi : (x_1, x_2) \mapsto (1, \sqrt{2}x_1, \sqrt{2}x_2, x_1^2, \sqrt{2}x_1x_2, x_2^2)$$

- Dot product in transformed space

$$\begin{aligned}\varphi(x) \cdot \varphi(z) &= 1 + 2x_1z_1 + 2x_2z_2 + x_1^2z_1^2 \\ &\quad + 2x_1x_2z_1z_2 + x_2^2z_2^2 \\ &= (1 + x_1z_1 + x_2z_2)^2\end{aligned}$$

- Transformed dot product can be expressed in terms of original inputs

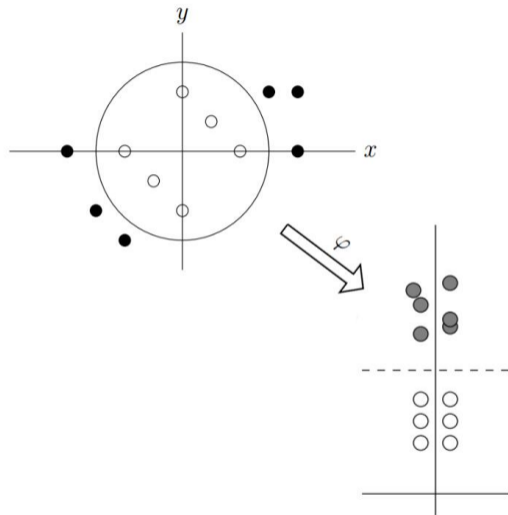
$$\varphi(x) \cdot \varphi(z) = K(x, z) = (1 + x_1z_1 + x_2z_2)^2$$



Kernels

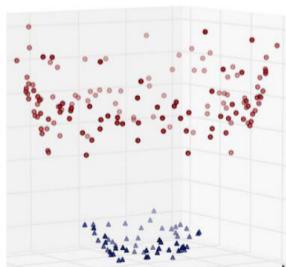
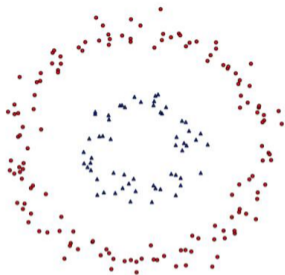
- K is a **kernel** for transformation φ if $K(x, z) = \varphi(x) \cdot \varphi(z)$
- If we have a kernel, we don't need to explicitly compute transformed points
- All dot products can be computed implicitly using the kernel on original data points

$$\text{sign} \left[\sum_{i \in \text{sv}} y_i \alpha_i (\varphi(x_i) \cdot \varphi(z)) + b \right]$$
$$\text{sign} \left[\sum_{i \in \text{sv}} y_i \alpha_i K(x_i, z) + b \right]$$



Kernels

- If we know K is a kernel for some transformation φ , we can blindly use K without even knowing what φ looks like!
- When is a function a valid kernel?
- Has been studied in mathematics — **Mercer's Theorem**
 - Criteria are non-constructive
- Can define sufficient conditions from linear algebra

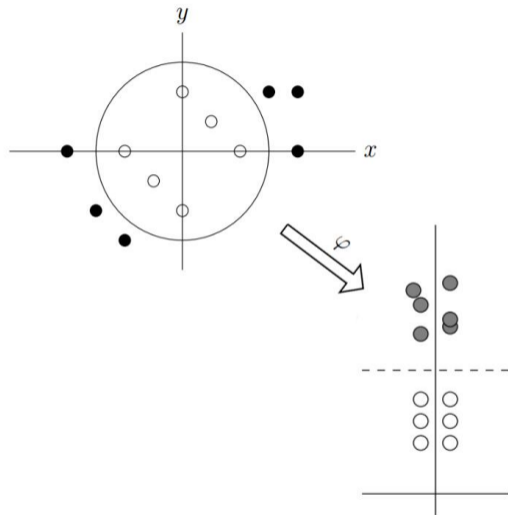


Kernels

- Kernel over training data x_1, x_2, \dots, x_N can be represented as a **gram matrix**

$$K = \begin{matrix} & x_1 & x_2 & \cdots & x_n \\ \begin{matrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{matrix} & \left[\begin{array}{cccc} & & & \\ & & & \\ & & & \\ & & & \\ & & & \end{array} \right] \end{matrix}$$

- Entries are values $K(x_i, x_j)$
- Gram matrix should be **positive semi-definite** for all x_1, x_2, \dots, x_N



Known kernels

- Fortunately, there are many known kernels
- Polynomial kernels
$$K(x, z) = (1 + x \cdot z)^k$$
- Any $K(x, z)$ representing a similarity measure
- Gaussian radial basis function — similarity based on inverse exponential distance

$$K(x, z) = e^{-c|x-z|^2}$$

